Energy-efficient computing for HPC workloads on Heterogeneous Chips

Akhil Langer, Ehsan Totoni, Udatta Palekar*, Laxmikant (Sanjay) V. Kale
Parallel Programming Laboratory, Department of Computer Science
*Department of Business Administration
University of Illinois at Urbana-Champaign
http://charm.cs.uiuc.edu/research/energy

PMAM 2015
6th International Workshop on Programming Models and Applications for Multicores and Manycores
February 7-8, 2015
Outline

- Introduction
- Background
- Problem Statement
- Approach
- Results
Introduction

• Motivation
  – Huge energy consumption of data centers
  – 20MW power @ $0.15 per KWh, costs $2.2 M per month
  – Energy efficiency identified as a major exascale challenge by DoE
  – Consider charging users in energy units (KWh) instead (or in addition) of SUs
Introduction

• Low voltage operation
  – For high energy efficiency
  – For example, 10x increase in energy efficiency near threshold voltage

• But
  – Variation in CMOS manufacturing process
  – Low voltage operation introduces variability on chip
  – Cores have different frequencies and power consumption
Process Variation

- Low voltage operation
Programming Systems*

• **Problem**
  – HPC applications are highly synchronized
  – Speed determined by speed of slowest processor

• **Solution**
  – Do overdecomposition of work (e.g. Charm++)
  – Load Balance according to core speeds

• **Result**
  – Overdecomposition ratio of 16 => 2-6% load imbalance
  – No changes required in application code

*Under Review
Problem Statement

• Not optimal to use all cores on chip for execution
  – Shared resources cause contention
  – High energy consumption
• A configuration is defined as the cores on which the application is run

Determine optimal configuration that minimizes energy consumption (with optional timing constraints) of the chip for a given application
Performance Modeling*

- Exhaustive evaluation of configurations infeasible

**Model 1**

- Sum of individual core performance
- Memory contention not modeled

\[ S = \sum_{i \in c} s_i \]

**Model 2**

- Add memory access time
- # of active cores not accounted

\[ T = \frac{T_{cpu}}{\sum_{i \in c} f_i} + T_{mem} \]

*Under Review
Performance Modeling*

- Model 3
  - One model each for configurations with same number of cores
  - Performance is linear function of frequency
  - Total #cores (n) models
    - k is number of cores in configuration c
    - $a_k$, $b_k$ are line constants
    - $f_i$ is frequency of core i
  - Average prediction error less than 1.6%
  - Dynamic power consumption can be modeled in same way

\[
S = a_k \left( \sum_{i \in c} f_i \right) + b_k
\]

*Under Review
Energy Optimization Approach

\[ \min \quad \sum_{k=1}^{N} (n_k \times (a_k^p \sum x_i f_i + b_k^p + \sum s_i x_i) \times (a_k^t \sum x_i f_i + b_k^t)) \]

**Select One Value of** \( k \)

\[ \sum_{k=1}^{n} n_k = 1 \]

**Total Number of Cores Equals** \( k \)

\[ \sum_{i=0}^{n-1} x_i = \sum_{k=1}^{n} n_k k \]

**Variables Range**

\[ \forall i \in [0, n), \quad x_i \in \{0, 1\} \]
\[ \forall k \in (0, n], \quad n_k \in \{0, 1\} \]

**Cubic Objective Function!**
Energy Optimization Approach

- Convert cubic program to n quadratic programs
- Each corresponding to all configurations with fixed number of cores
- Select best configuration across n quadratic programs

\[
\begin{align*}
\text{min} & \quad \left( a^p_K \sum_{i=0}^{n-1} x_i f_i + b^p_K + \sum_{i=0}^{n-1} s_i x_i\right) \\
\text{s.t.} & \quad \left( a^t_K \sum_{i=0}^{n-1} x_i f_i + b^t_K\right) \\
\sum_{i=0}^{n-1} x_i &= K \\
\forall i &\in [0, n), \quad x_i \in \{0, 1\}
\end{align*}
\]

Total Number of Cores Equals \( K \)

Static Power

Dynamic Power

Variables Range

Quadratic Objective Function!
Energy Optimization Approach

- Quadratic programs hard to solve using non-linear methods
- Replace quadratic terms of form $x_1 x_2$ with binary variables $y_{12}$ and add following constraints
  \[
  y_{12} \leq x_1 \\
  y_{12} \leq x_2 \\
  y_{12} \geq x_1 + x_2 - 1
  \]
- Add timing constraint
  \[
  a_K^t F + b_K^t \leq P t_{min}
  \]
  where $F$ is sum of frequencies, and $P$ is allowed time penalty
Setup

• **Sniper Simulator**
  – \( V_{dd} = 0.765V \)
  – 36 cores on chip
  – Results across 25 chips

• **Applications**
  – miniMD
    • Molecular dynamics mini application
    • Computationally intensive
  – Jacobi
    • 3D stencil code
    • Memory intensive

• **Heuristics**
  – Min heuristic
  – Max heuristic

• **Integer Linear Program (ILP) Solver**
  – Gurobi
  – Uses variant of branch-and-bound method
Results

Figure 3: Percentage savings in energy with MIN, MAX heuristics, and the ILP method for the two applications, miniMD and Jacobi3d, with respect to the configuration with best execution time. The bars correspond to the average benefits, while the vertical lines correspond to the minimum and maximum benefits obtained from the corresponding heuristic across the 25 chips. In (a), configuration that minimizes energy consumption is sought irrespective of penalty in execution time of the application. In (b) and (c), the best configuration that minimizes energy while the execution time penalty is less than 15% and 5%, respectively, is sought using the MIN, MAX heuristic, and ILP method.

Figure 4: An example of a configuration selected by the ILP optimization method for Jacobi application. Circle markers correspond to the cores selected by ILP. A total of 21 cores were selected by the ILP method. MIN, MAX heuristic selected 26, 27 cores, respectively.

6.2 Solution Time

The proposed methodology requires optimizing n^2 proper ILPs, where n is the total number of cores on the chip. Each ILP has 702 variables, and 2000 constraints. The ILP optimizations are independent of each other and can therefore be very easily parallelized by launching them in parallel on multiple cores of a compute node and/or on multiple compute nodes. For the experiments, we use a Dell 2.67 GHz Dual Westmere Xeon E5640 processor with a total of 8 cores and 16 SMT threads. The ILP optimizations required for a given chip and an application were launched in parallel on the machine. We now consider the solution time for each of the three cases presented in Section 6.1.

1. When there is no execution time penalty constraint, it took an average of 400 seconds and 1090 seconds for obtaining the optimal result for miniMD and Jacobi, respectively. An average of 4.08e7 and 3.27e8 simplex iterations (summed across all the BnB vertices explored) were performed by the ILP solver for miniMD and Jacobi, respectively.

2. When the maximum execution time penalty of 15% is enforced, the configuration search space for ILP optimization is reduced significantly. It took an average of 14.8s, 37s to find the optimal solution for miniMD, Jacobi, respectively.

3. With the maximum execution time penalty of 5%, the search space is further reduced, and it took only 8s, 10.2s to find the optimal solution for miniMD, Jacobi, respectively.

We compare these results with exhaustive evaluation of the performance and power models for all the configurations on the same machine. The configurations can be evaluated in parallel. The total number of configurations to be evaluated...
Energy Savings

(a) With no time constraint

26%

(b) Maximum 15% time penalty

18.4%

(c) Maximum 5% time penalty

13.4%

ILP Solution Time:

745 seconds

26 seconds

9 seconds

vs

Exhaustive Evaluation: 74 hours
Conclusions

• Negligible overhead
  – $O(n)$ samples required
  – Performance models developed with negligible overhead

• ILP solvers to optimize energy consumption with timing constraints
  – Significant energy savings as compared to sub-optimal heuristics

• No extra compute resources required
  – Solve ILPs on respective chips prior to job execution
Future Work

• Further improvement of performance models
• Evaluate approach with even larger number of cores
• Optimization methods to further improve solution time
• Apply to other HPC applications
QUESTIONS!

http://charm.cs.uiuc.edu/research/energy

Energy-efficient computing for HPC workloads on Heterogeneous Chips

Akhil Langer, Ehsan Totoni, Udatta Palekar*, Laxmikant V. Kale
Parallel Programming Laboratory, Department of Computer Science
*Department of Business Administration
University of Illinois at Urbana-Champaign

PMAM 2015
6th International Workshop on Programming Models and Applications for Multicores and Manycores
February 7-8, 2015
San Francisco Bay Area, USA